UNIVERSITY OF MACEDONIA

SCHOOL OF INFORMATION SCIENCES

DEPARTMENT OF APPLIED INFORMATICS

PLANNING AND SCHEDULING (2nd semester)

Work: 3rd Work
Academic Year: 2024-2025
Speaker: Ilias Sakellariou

Exercise 1: Placing Facilities

On a map, you need to place a number of service points (facilities) so as to minimize the distance from predetermined points where your customers are located. The map square grid with a given dimension, and we consider it to have integer coordinates. *All distances listed below are Manhattan distances* . The problem data are:

The size of the map (i.e. we have a 25x25 area):

```
grid_size = 25;
```

On this map the coordinates start at (1,1) and go up to (25,25). These are the points where both customers and service points enter.

The number of customers

```
num_clients = 3;
```

The customer locations are given by the following 2-dimensional array:

```
client_loc = [|4,15|12,4|23,15|];
```

Obviously, (4,15) are the coordinates on the map of the first customer, (12,4) of the next, and so on.

The number of service points:

```
num_facilities = 2;
```

Each service point has a (different) minimum distance that it must maintain from each customer. The minimum distances are given in the following table :

```
min_distance = [ 4 ,3 ];
```

In the table above, the minimum distance of any customer from the first service point must be *strictly greater* than 4.

Service points must have a minimum distance between them:

```
distance_between_facilities = 3;
```

The above means that the distance between any two service points must be *strictly greater* than 3.

(a) What is required is a "fair" distribution, i.e. minimizing the distance of each customer from their nearest service point (minimizing the max of mins). Implement a program in MiniZinc that finds the solution (in the **client_min_distance** table) with the minimum total cost (variable **max_of_mins**). Example execution:

```
client_min_distance = [7, 4, 12];
max_of_mins = 12;
----------
```

In the above solution, the distance of the first customer from its closest point is 7, the second is 4, etc. The max_of_mins is 12.

(b) Try different heuristics (at least 3) with the gecode solver, on the different problems given to you. Finally, try the same code with the OR Tools CP SAT solver, for all the problems. Report your results in the report you will submit.

You can set an upper limit on the time you will allow the solver to find a solution.

Note : The solution shown is NOT optimal.

You will find the necessary files in eclass.

The biggest problem is that it can take a long time to execute.

Exercise 2 – Container Delivery

In a port there are N derricks that load boxes onto transport vehicles for delivery to one of two warehouses. Each derrick takes 4 moments to load a box onto a truck. The trucks have to handle orders with a different number of boxes each.

In each warehouse, there is only one mobile crane (reach stacker) that can unload one box at a time. The available vehicles should:

1. load the order from the port,
2. go to the warehouse
3. and at some point after their arrival unload the order.

(a) The relevant information is given in tables in the corresponding data files (MiniZinc data files), such as:

the number of cranes in the port:

```
num_derricks = 3;
```

the number of vehicles

```
num_trucks = 4;
```

The order that each vehicle will handle and the warehouse to which it will be delivered:

```
delivery =     [|5,enum2int(wh1)
                |2,enum2int(wh2)
```

```
                    |3,enum2int(wh1)
                    |4,enum2int(wh2) |];
```

For example, the first vehicle must transport 5 boxes to warehouse wh1, the second 2 to warehouse wh2, kok.

In the model you also have the following information:

```
enum WAREHOUSES = {wh1,wh2};
array[WAREHOUSES] of int: travel_time = [15,20];
```

the first is the enumerated type with the symbolic names of the warehouses, and the corresponding transit times from the port to the warehouse (15 for wh1).
Write a program in MiniZinc that calculates

- **StartLoad** when will the loading of each vehicle at the port begin,
- **StartUnload** when the unloading of each vehicle in the warehouse will begin,
- **End** , when will the unloading of each vehicle be finished,
- **makespan** is the completion time of the entire program, which should be as short as possible (shown as _objective below).

For example:

```
StartLoad = [0, 12, 0, 0];
StartUnload = [35, 40, 27, 36];
End = [40, 42, 30, 40];
_objective = 42;
```

where in the example above vehicle 1 will start loading the boxes at time 0, and unloading at time 35, while the total schedule will last a total of 42 minutes.

The variables shown above are the ones necessary for the solution. You can define others if you wish.

Assume that the schedule starts at time 0 and must be completed in max_time minutes at most (the TIME set is already declared).

You will find the necessary files in eclass.

Notes

- The necessary files for the implementation of the two exercises can be found in ECLASS (Documents-> Constraints-> Assignments) **.**
  - Exercise 1: folder placing_facilities, project file facilityPlacement.mzp
  - Exercise 2: port_delivery folder, portDelivery Project.mzp project file

SURRENDER

You will deliver within the date indicated in ECLASS ONE **zip FILE** with the following (zip NOT 7z, rar, etc.):

- The above files **with the name and directory structure** they have in ECLASS **, which will contain your solutions.**
- **report.pdf** file (***in pdf format***), in the original zip folder, which will contain:
  - **On the first page, your Name, your Registration Number and your email.**

- For each of the exercises:
    - code (regardless of whether **it** is in the corresponding file) and a comment about it.
    - Examples of execution (1-2 where possible)
    - Bugs and problems your code has.

ATTENTION : STRICTLY FOLLOW THE VARIABLE NAMES GIVEN ABOVE (AUTOMATIC CODE CHECK)

Good luck (*and have fun with MiniZinc!*)